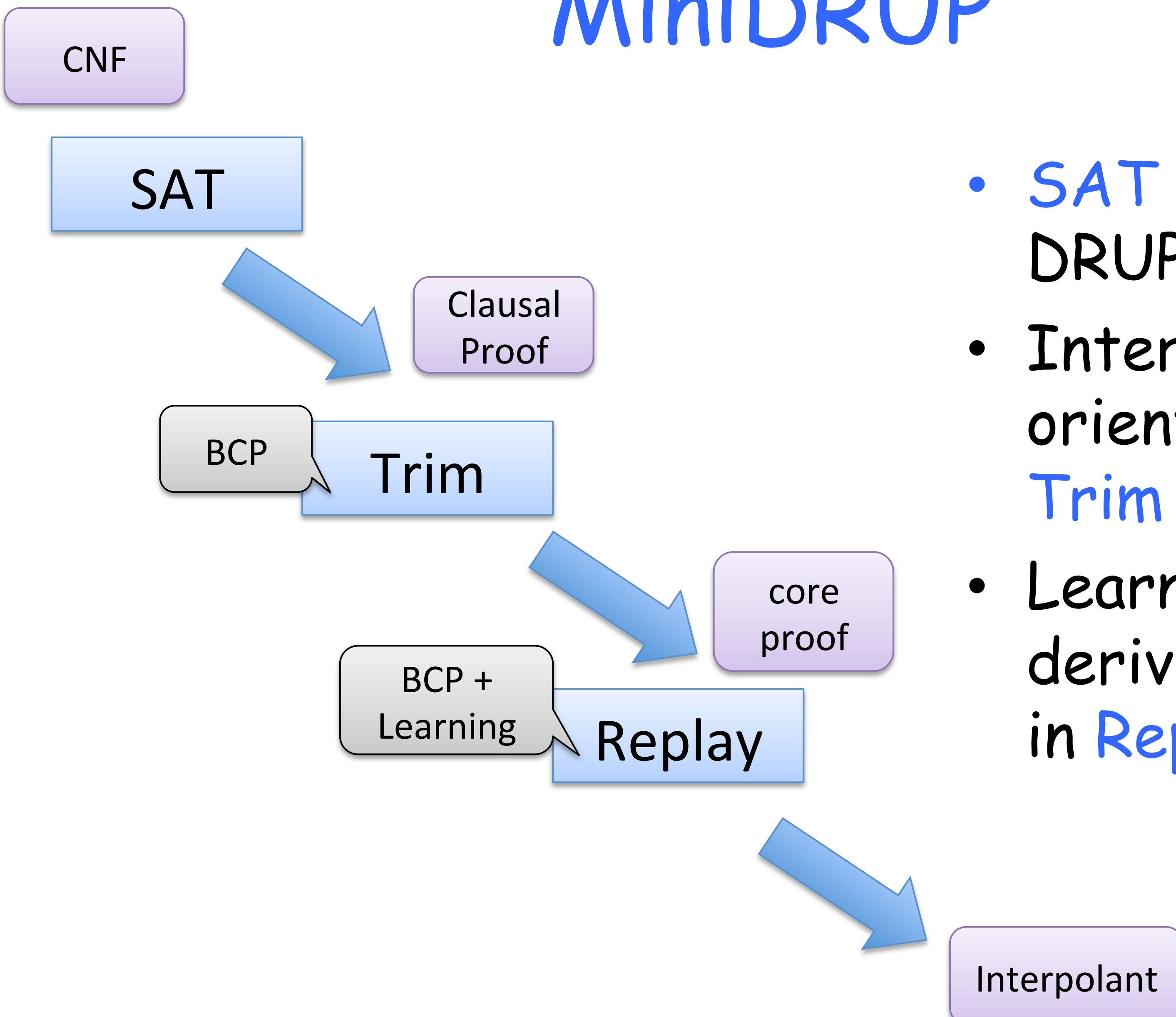
Interpolants from Clausal Proofs

Arie Gurfinkel[1]
Yakir Vizel[2]

iPRA 2014
Vienna, Austria

1. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, USA
2. Computer Science Department, Technion, Israel

MiniDRUP



- SAT with DRUP proofs
- Interpolation-oriented BCP in Trim
- Learn shared-derived clauses in Replay

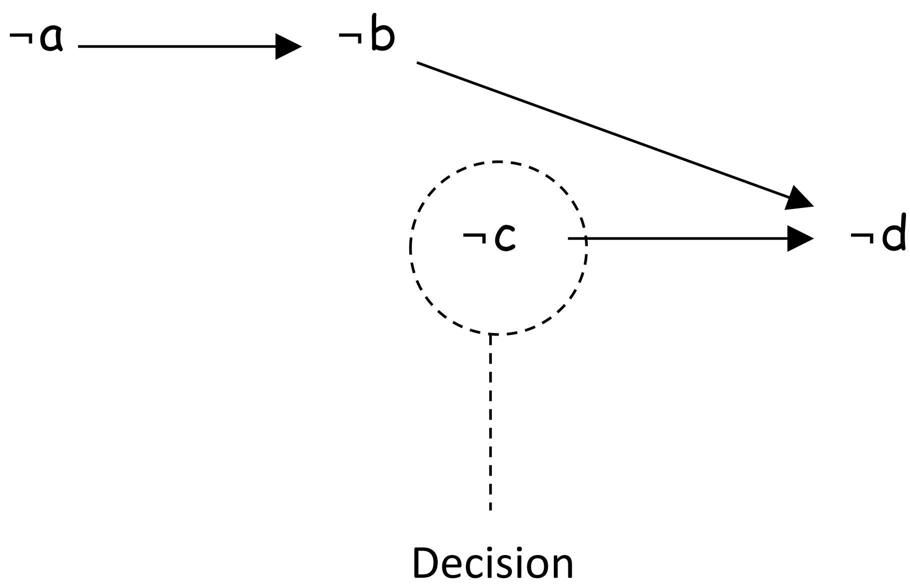
CDCL SAT solvers

- Check satisfiability of a CNF formula
  - CNF is conjunction of clauses and
  - Clause is a disjunction of literals
- Basic steps:
  - Arbitrary decisions for un-assigned vars
  - Propagate values (BCP)
  - Analyze conflicts and change decisions

SAT solvers can generate refutation proofs

The Implication Graph (BCP)

$$\neg a \wedge (a \vee \neg b) \wedge (b \vee c \vee \neg d)$$



Propositional Resolution

$$\begin{tikzcd} a \lor C \arrow[dr] & & \neg a \lor D \arrow[dl] \\ & C \lor D & \end{tikzcd}$$

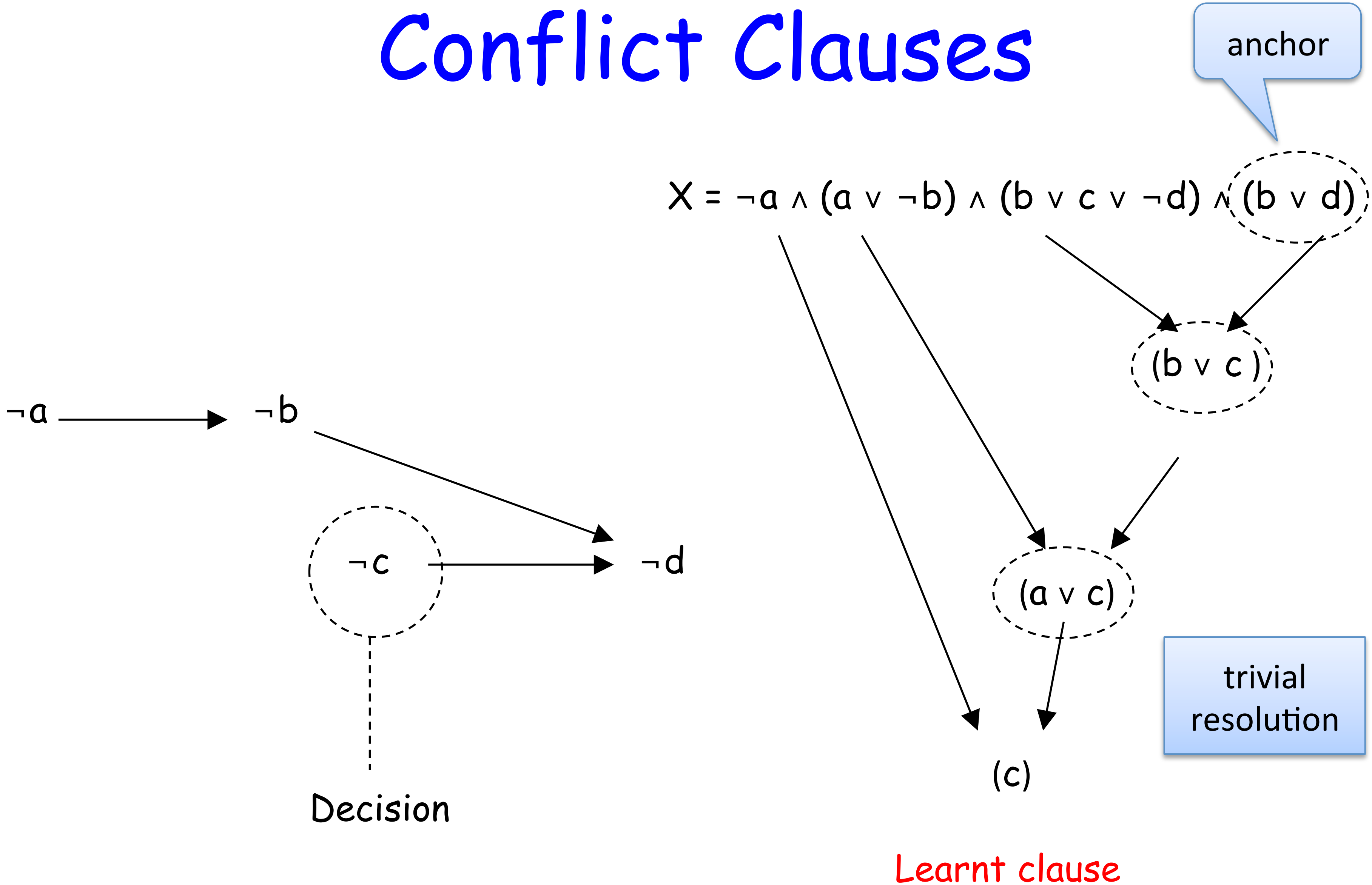
Analyzing a Conflict

- Decisions made by the SAT solver may lead to a conflict
  - A clause is evaluated to false under the current assignment
- The implication graph is used to guide resolution steps
- The result is a learnt clause
  - Prevents the same conflict from re-appearing

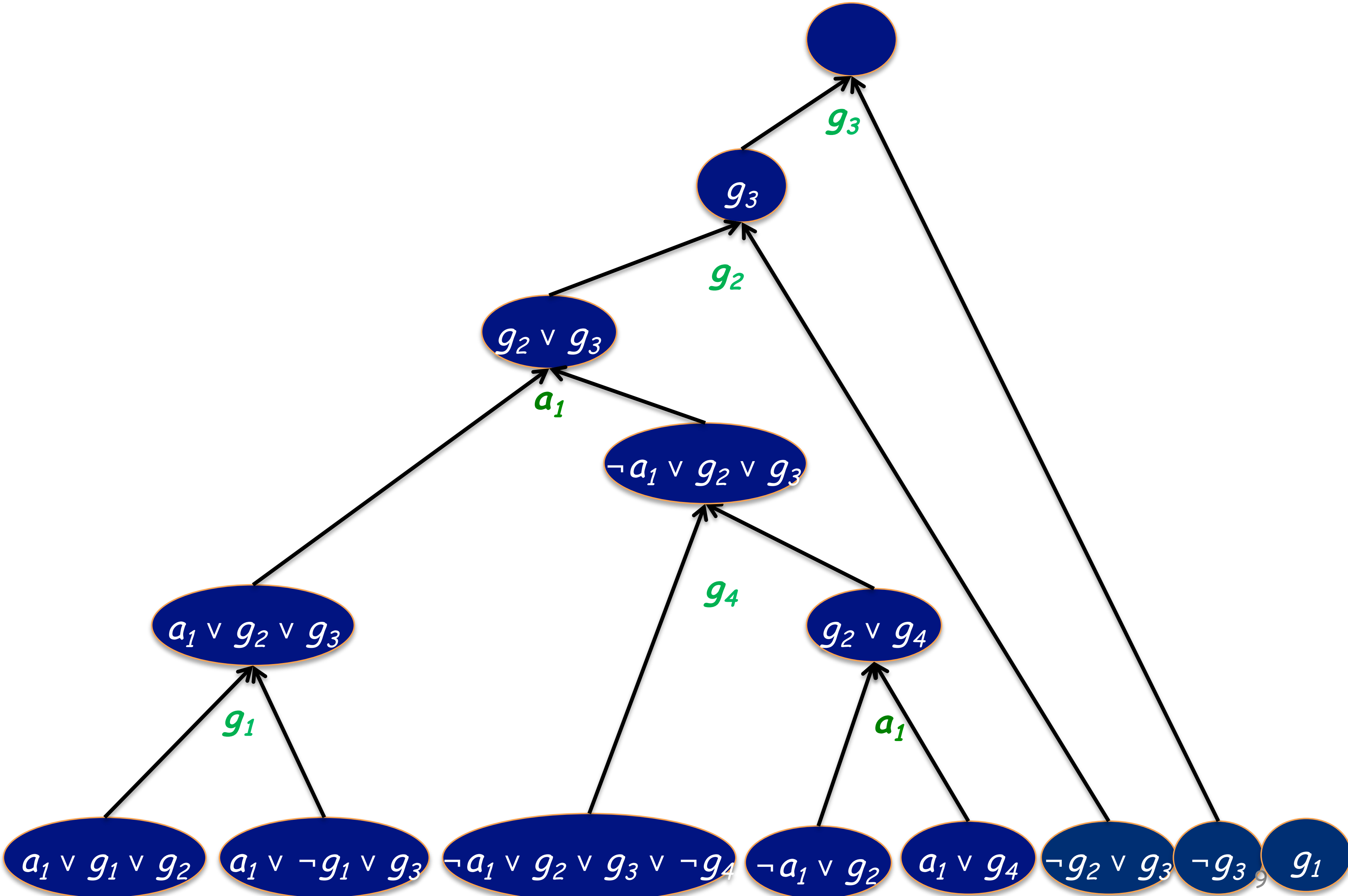
Refutation Proofs

- A formula is UnSAT when the empty clause can be derived from the original formula
- Resolution proof
  - A DAG that tracks resolution steps leading from the original clauses to the empty clause
    - Leaves – original clauses
    - Intermediate nodes – learnt/derived clauses
- Clausal proof
  - A sequence of learnt clauses
    - In the order they are learnt

Conflict Clauses



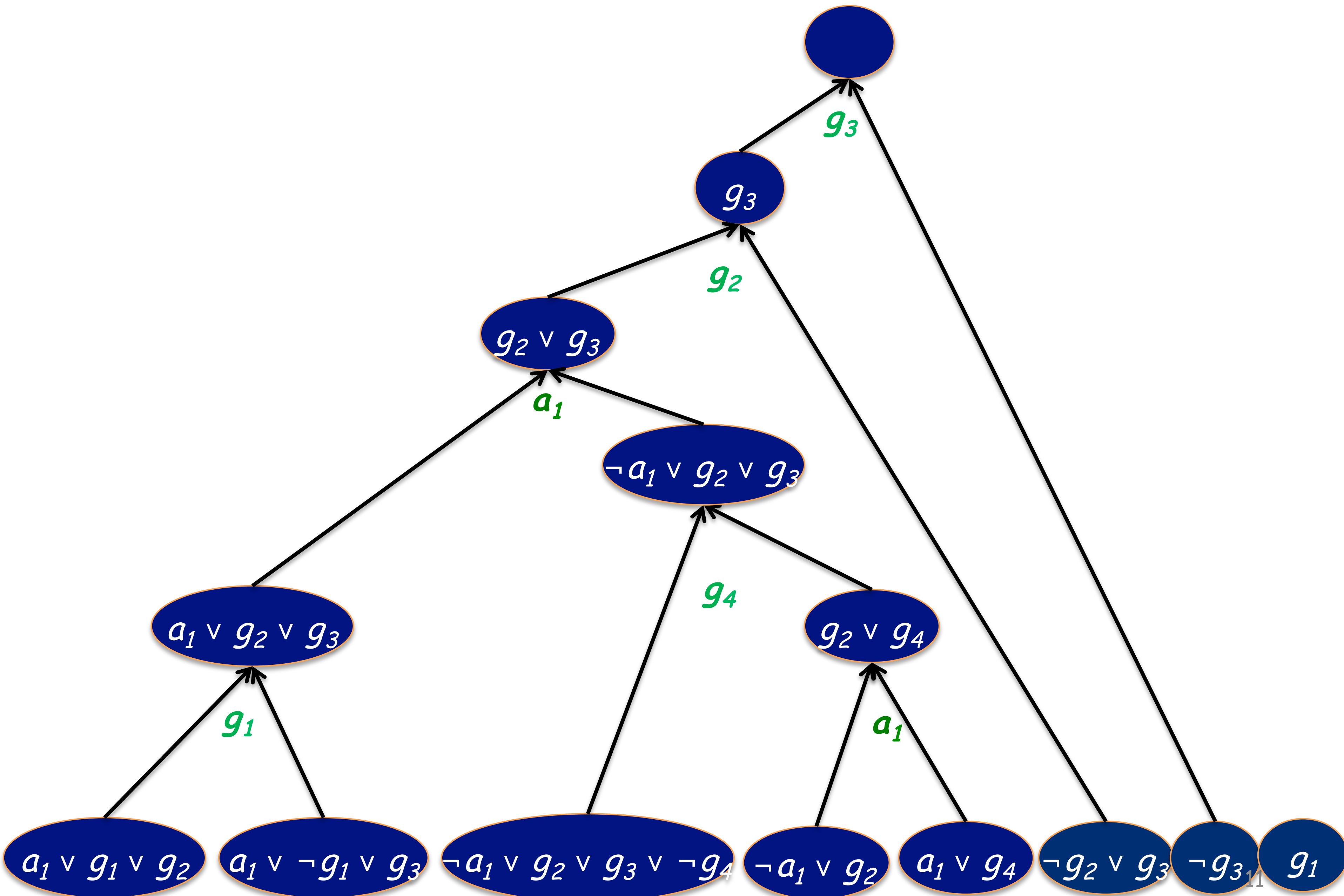
Resolution Proof



Clausal Proof

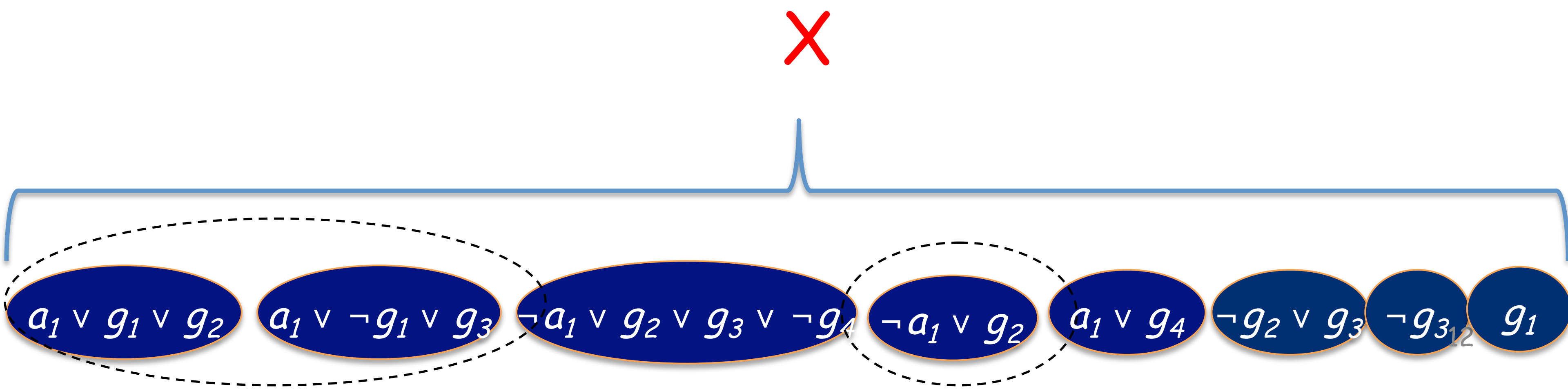
- Record learnt clauses in the order they are learnt
  - A learnt clause is derived by Trivial Resolution from some previous clauses
    - If prior to learning c, the CNF is X, then c is derived by Trivial Resolution if running BCP on $X \wedge \neg c$ leads to a conflict
- for our example, clausal proof is $\langle X, c \rangle$

Clausal Proof



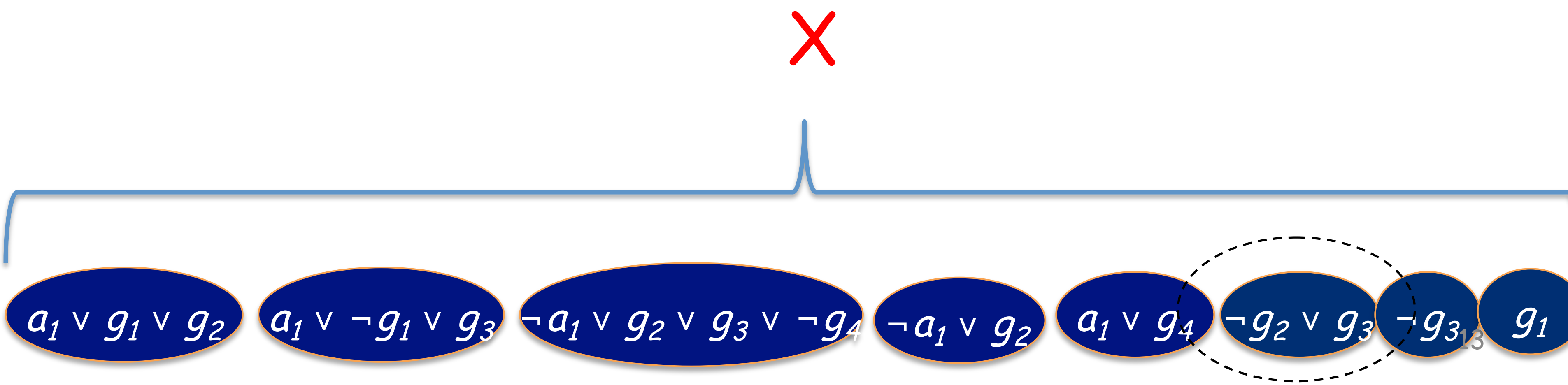
Clausal Proof

- $\langle X, (g_2 \vee g_3), (g_3) \rangle$
- $X \wedge \neg g_2 \wedge \neg g_3$
  - $\neg a_1$
  - g_1 , $\neg g_1$ → conflict



Clausal Proof

- $\langle X, (g_2 \vee g_3), (g_3) \rangle$
- $X \wedge (g_2 \vee g_3) \wedge \neg g_3$
  - g_2
  - $\neg g_2$ → conflict

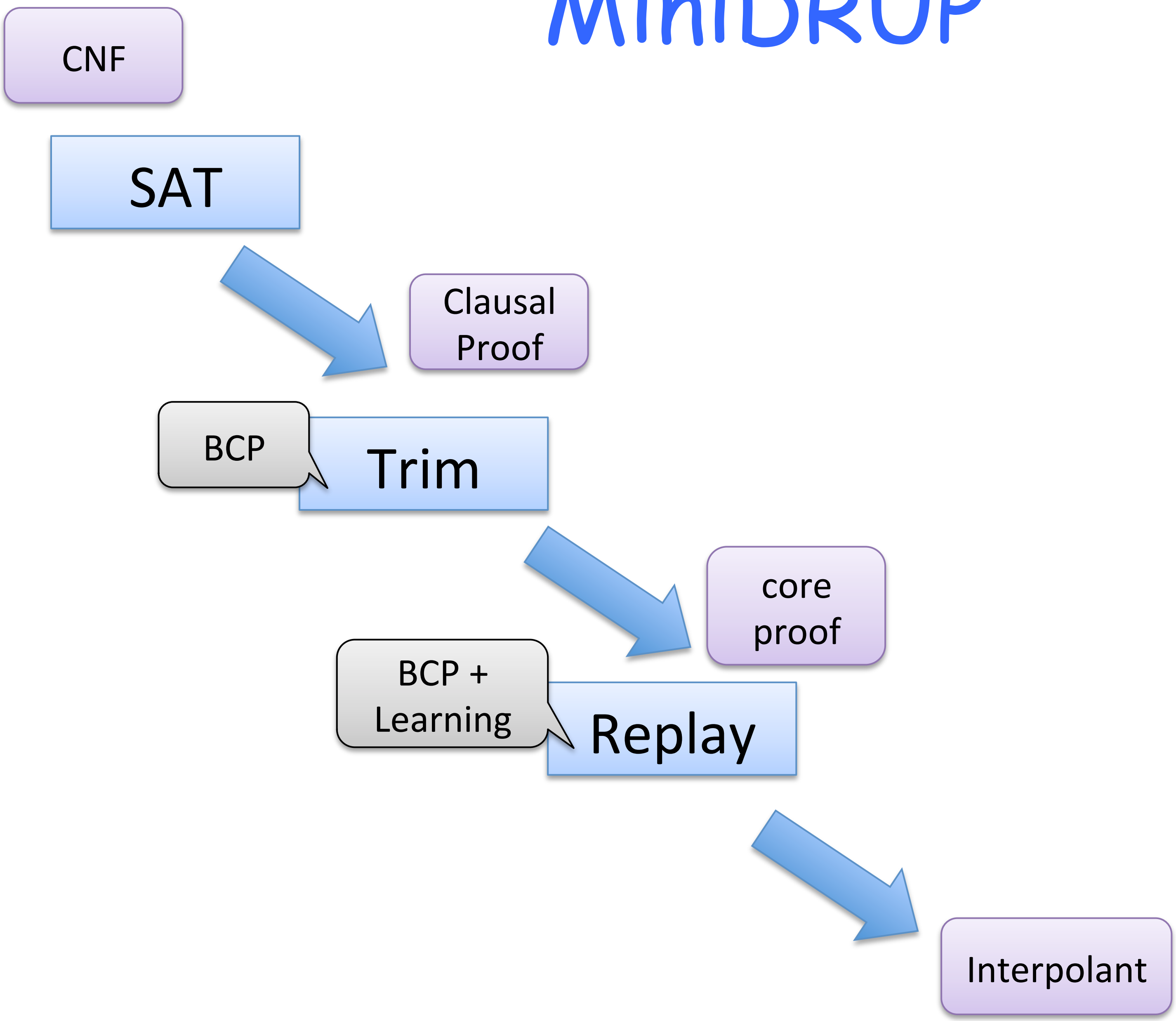


DRUP Proof

Marijn et al. FMCAD'13

- Extends a clausal proof by tracking deleted clauses
  - A SAT solver deletes learnt clauses
- $\langle X, c_1, c_2, c_3, c_2^*, c_4, c_1^*, c_3^*, \ldots \rangle$
  - Why?
- Introduced for SAT-solvers certification

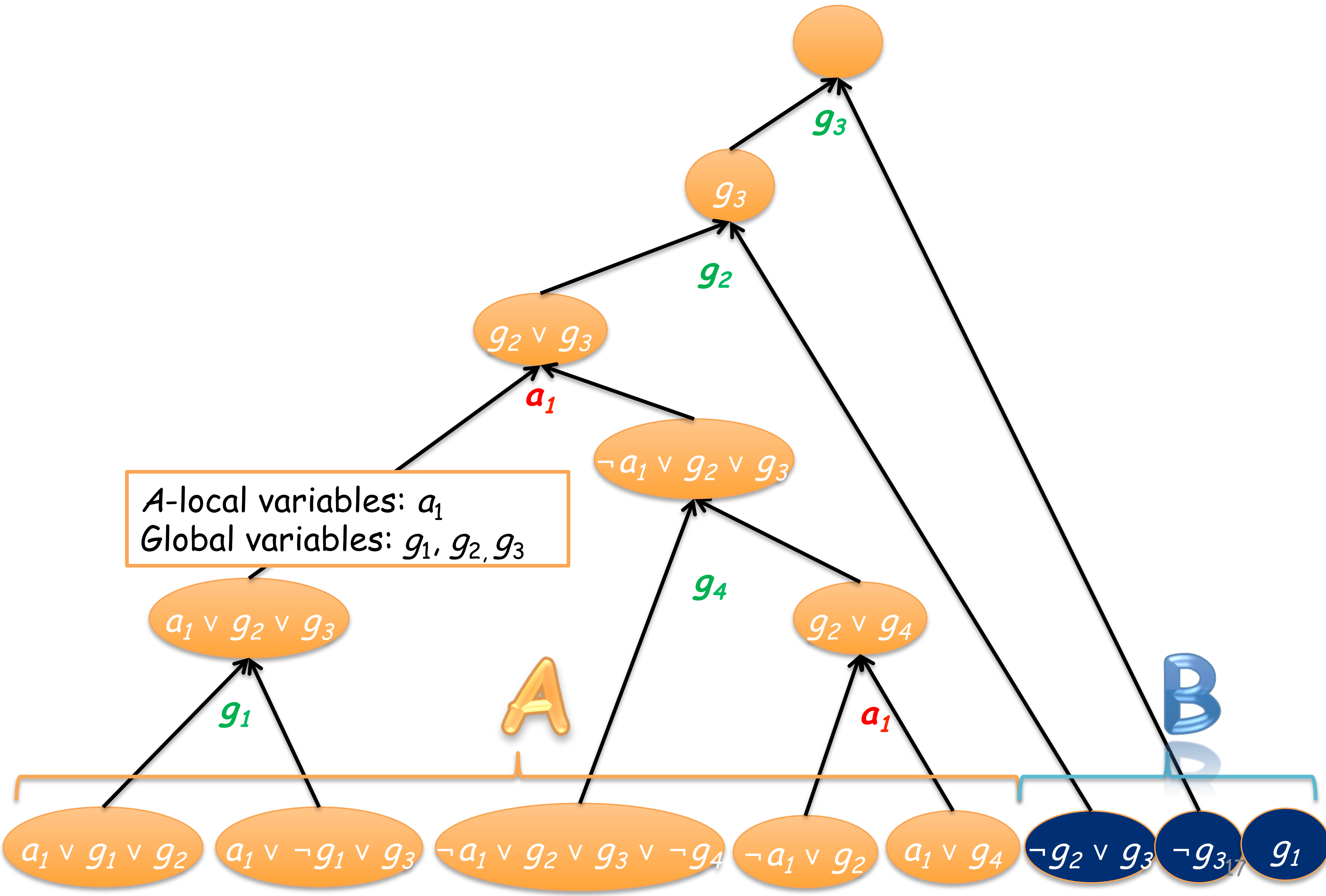
MiniDRUP


CNF
SAT
Clausal Proof
BCP
Trim
core proof
BCP + Learning
Replay
Interpolant

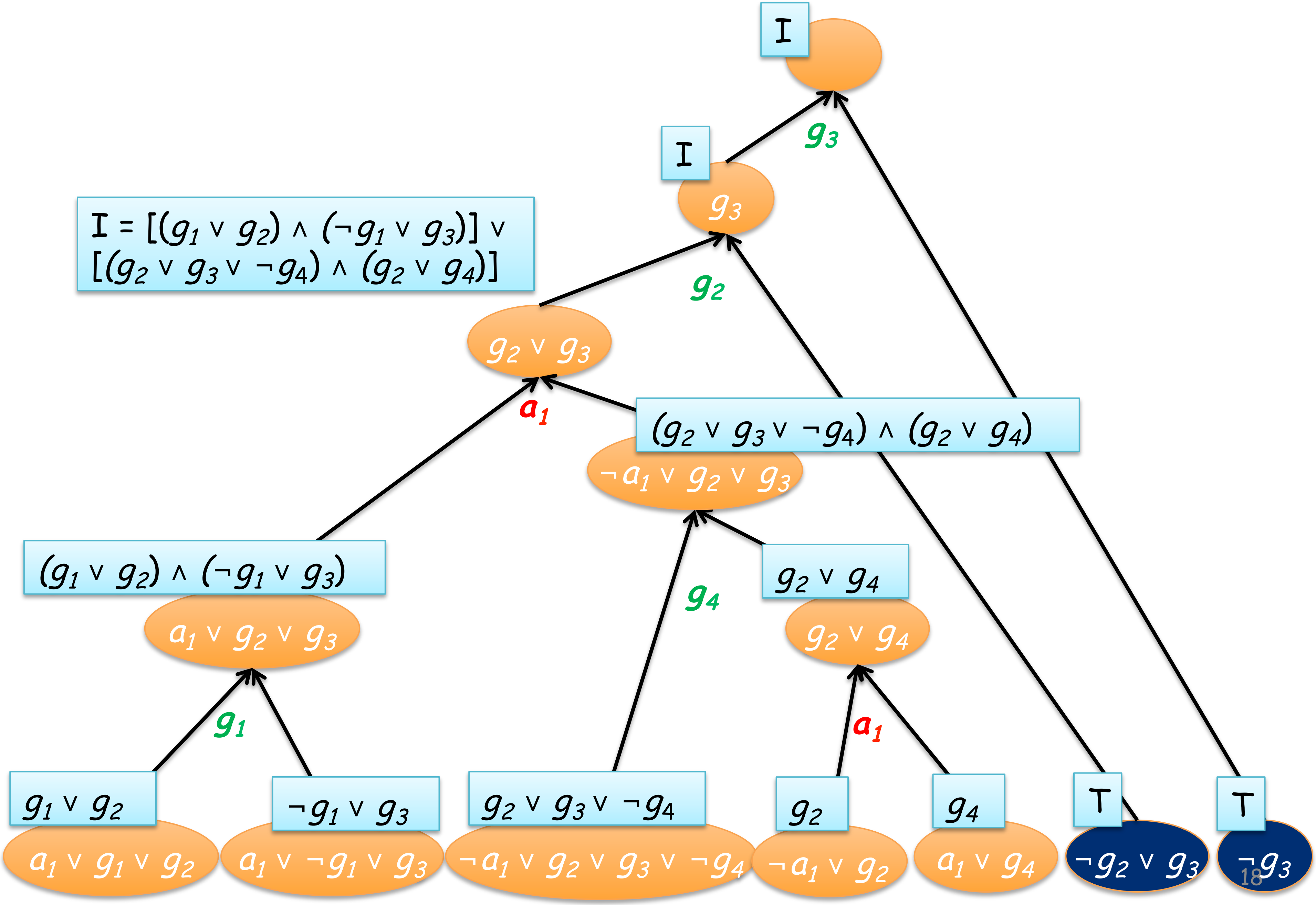
Interpolants

- Given an unsatisfiable pair (A,B) of propositional formulas
  - $A(X,Y) \wedge B(Y,Z)$ is unsatisfiable
- There exists a formula I such that:
  - $A \rightarrow I$
  - $I \wedge B$ is unsatisfiable
  - I is over the common variables of A and B

Resolution Proof



McMillan's Method

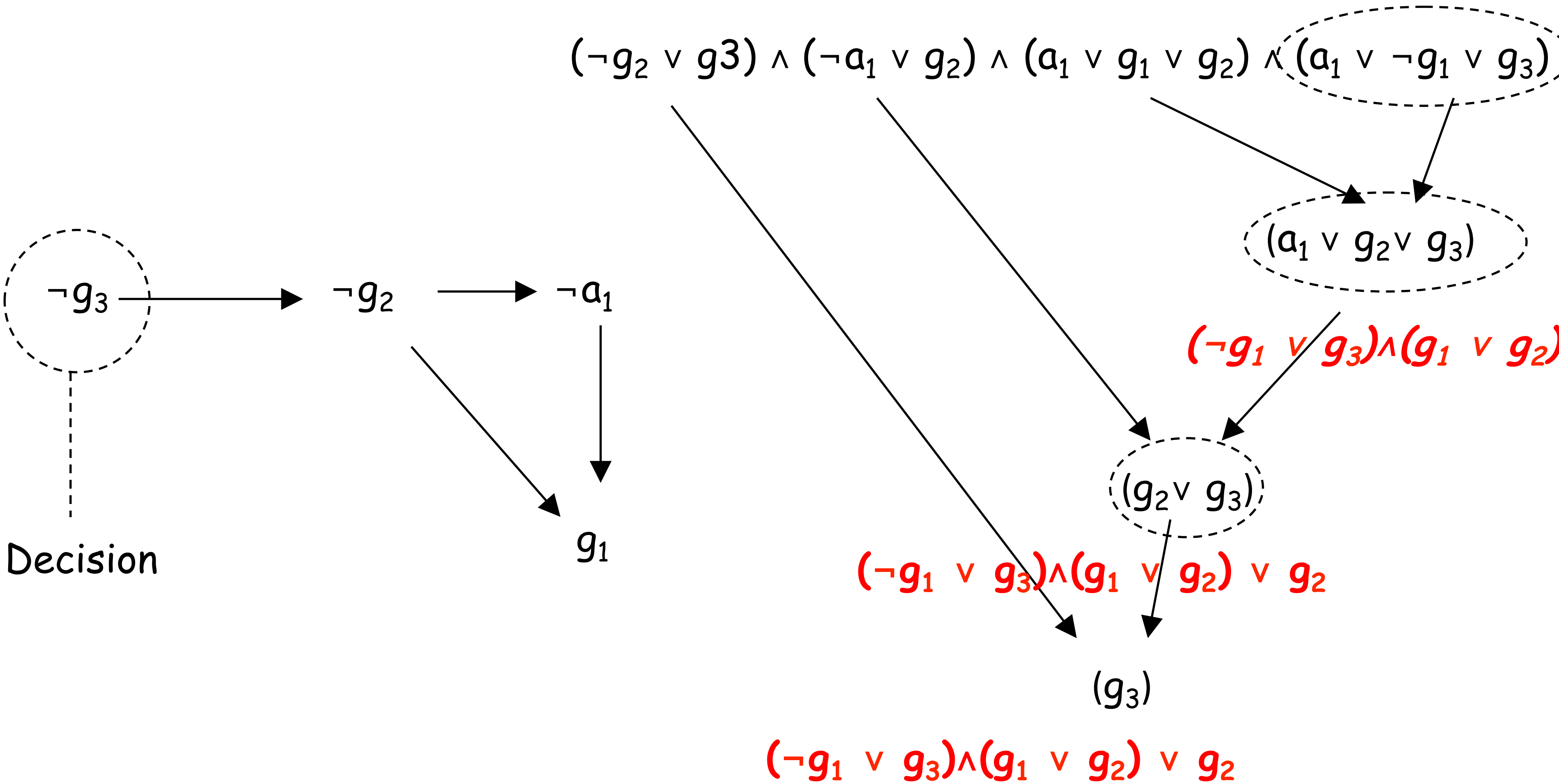


Clausal Proof

- $\langle X, (g_3) \rangle$
- $X \wedge \neg g_3$
  - $\neg g_2$
  - $\neg a_1$
  - $g_1 , \neg g_1$ → conflict

$$\overbrace{a_1 \vee g_1 \vee g_2 \quad a_1 \vee \neg g_1 \vee g_3 \quad \neg a_1 \vee g_2 \vee g_3 \vee \neg g_4 \quad \neg a_1 \vee g_2 \quad a_1 \vee g_4 \quad \neg g_2 \vee g_3 \quad \neg g_3 \quad g_1}^{X}$$

Conflict Clauses



Shared Derivable Clauses

- Given an unsatisfiable pair (A,B) of propositional formulas

- A clause c is shared-derivable iff
  - c is over the common variables of A,B
  - c is derived using only A clauses
    - Or, A => c

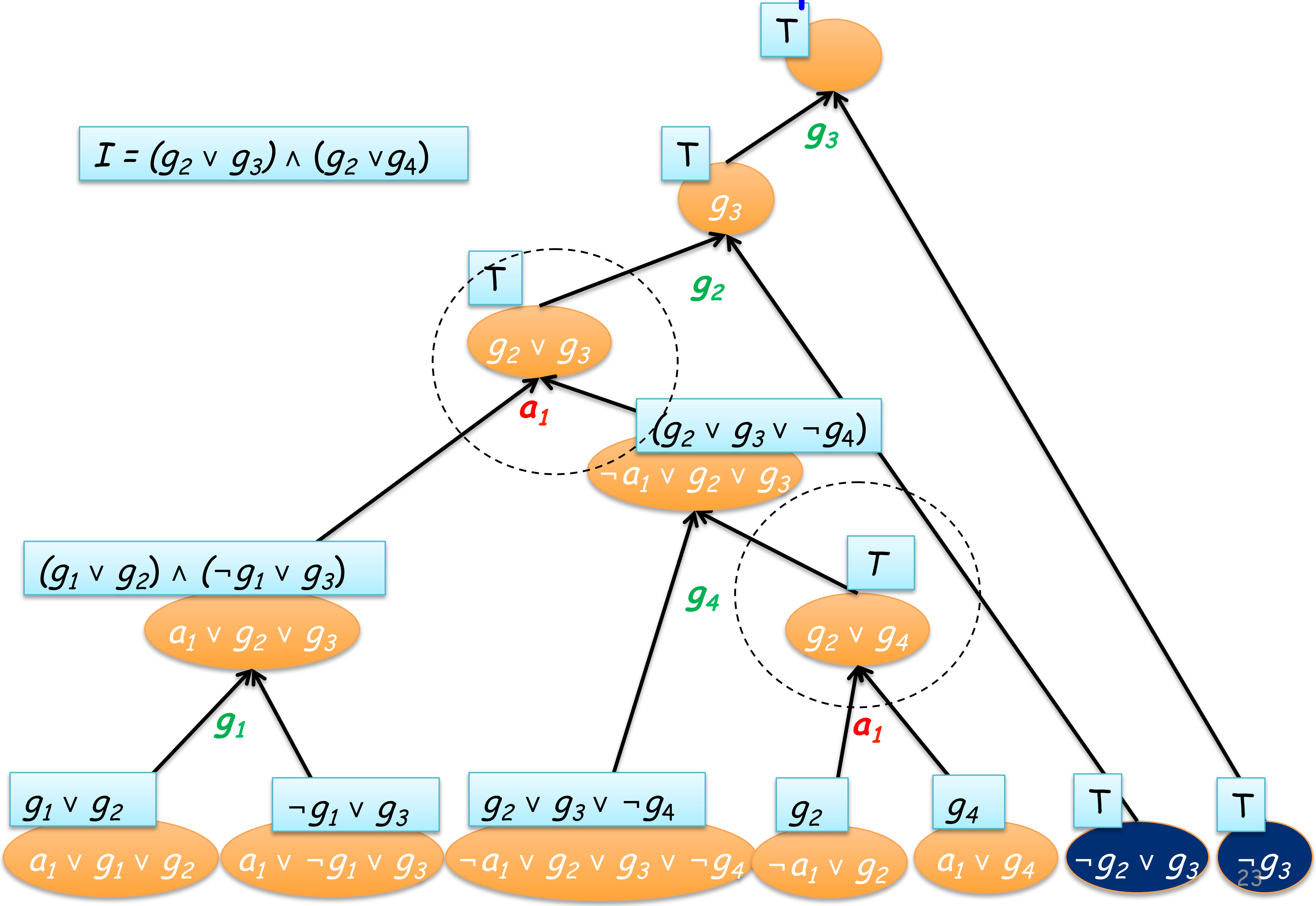
Partial CNF Interpolants

- Given an unsatisfiable pair (A,B) of propositional formulas

- Find shared-derivable clauses in the proof and
  - Log them as a CNF formula g
  - Treat them as B clauses during the computation

- Interpolant is $I \wedge g$

Partial CNF Interpolants



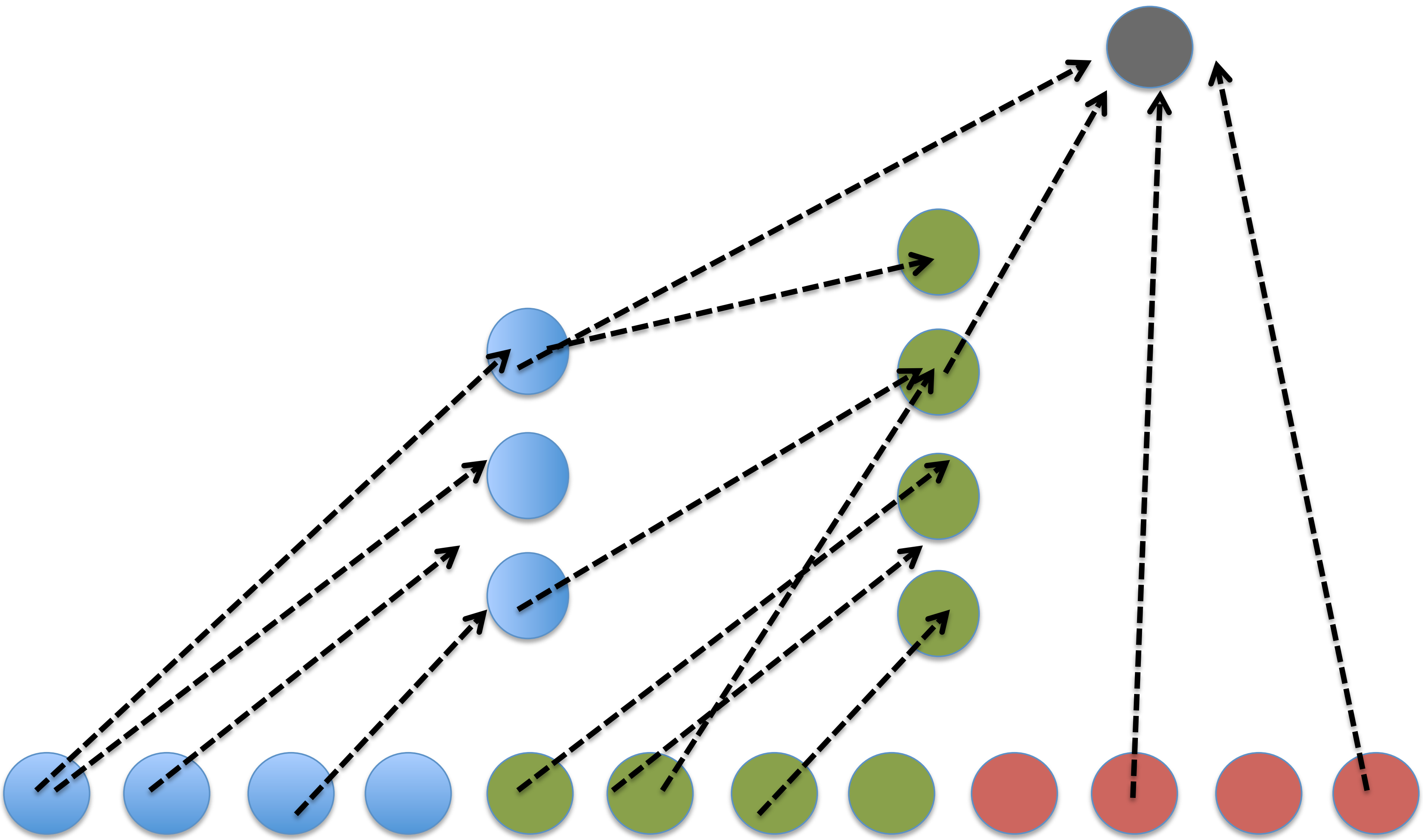
Sequence Interpolants

- Given an unsatisfiable tuple (A,B,C) of propositional formulas
  - $A(X,Y) \wedge B(Y,Z) \wedge C(Z,W)$ is unsatisfiable
- There exist formulae I_1, I_2 such that:
  - $A \rightarrow I_1$
  - $I_1 \wedge B \rightarrow I_2$
  - $I_2 \wedge C \rightarrow \mathrm{FALSE}$
  - I_1 is over the common variables of A and (B,C)
  - I_2 is over the common variables of (A,B) and C

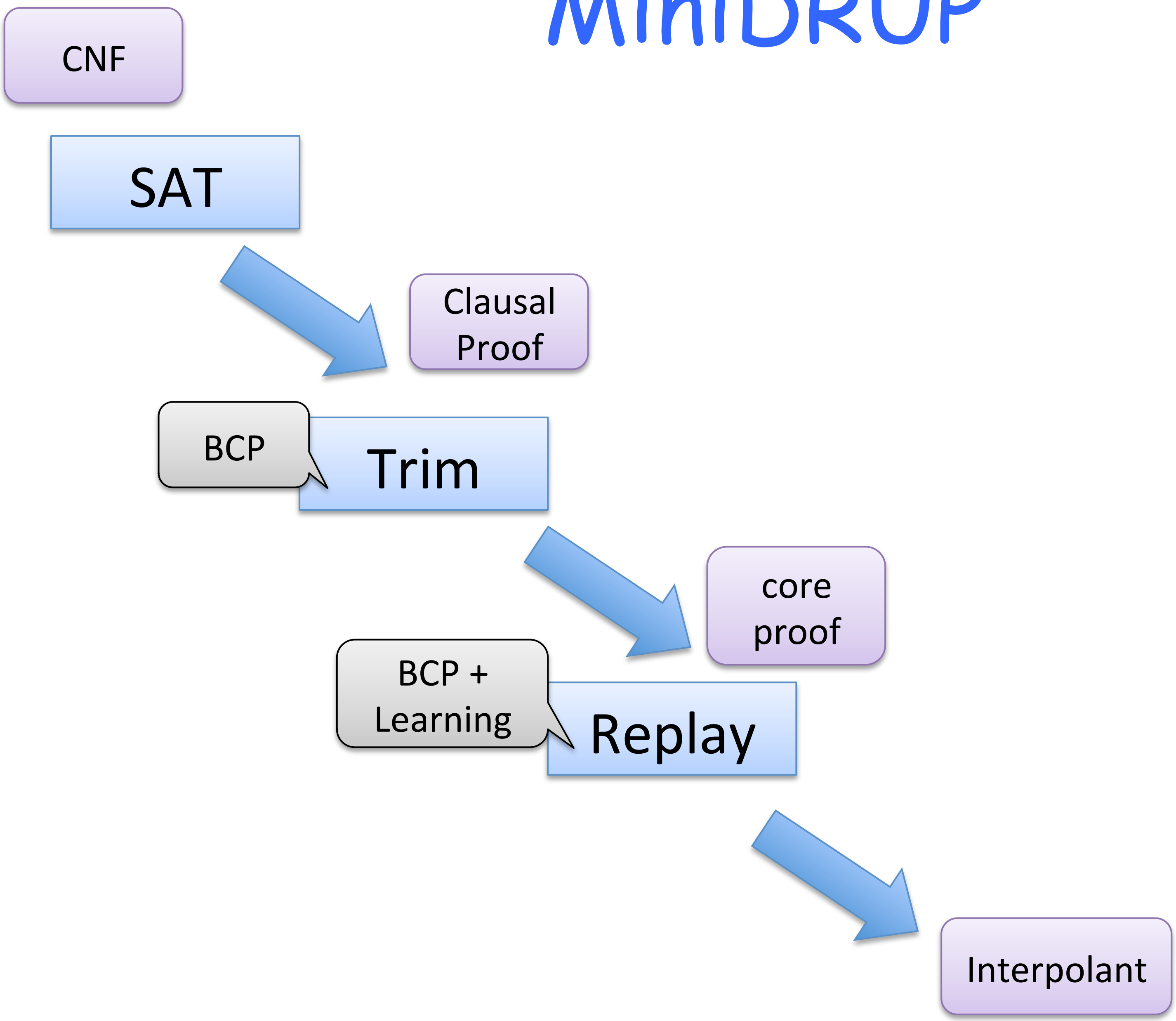
Sequence Interpolants

- A sequence of partial CNFs
  - It is more complex to maintain the sequence property
- A clause is shared-derivable iff:
  - It is derived using only shared-derivable clauses from previous partitions and from clauses within its own partition

Sequence Interpolants



MiniDRUP


CNF
SAT
Clausal Proof
BCP
Trim
core proof
BCP + Learning
Replay
Interpolant

Restructuring Proofs

- Proofs generally do not have this "special" structure
- Need to force this structure on the proof
  - CNF interpolants are exponentially weaker than general interpolants
  - Must be efficient
  - We do not want to disturb the SAT solver

Restructuring Proofs

- **Observation/Intuition** - let c be a clause over shared vocabulary then one of the following must hold:
  - c is shared-derivable
  - c can be derived using shared-derivable clauses

Experiments

Info

- Visit our web site
- http://arieg.bitbucket.org/avy/
- Come to our CAV talk…

Thank You